C 13

```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
```

```
_$2

Sym
---
NT$
NT$
NT$
NT$
NT$
NT$


NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$

NT$

NT$
NT$
NT$
NT$
NT$
NT$
NT$



NT$
NT$
NT$
NT$
NT$
NT$
PI
```

```
                                   E 6
**FILE**ID**RMOPRFLNM




          RRRRRRRR    MM      MM    000000    PPPPPPPP    RRRRRRRR    FFFFFFFFFF  LL          NN      NN  MM      MM
          RRRRRRRR    MM      MM    000000    PPPPPPPP    RRRRRRRR    FFFFFFFFFF  LL          NN      NN  MM      MM
          RR      RR  MMMM  MMMM  00      00  PP      PP  RR      RR  FF          LL          NN      NN  MMMM  MMMM
          RR      RR  MMMM  MMMM  00      00  PP      PP  RR      RR  FF          LL          NN      NN  MMMM  MMMM
          RR      RR  MM  MM  MM  00    0000  PP      PP  RR      RR  FF          LL          NNNN    NN  MM  MM  MM
          RR      RR  MM  MM  MM  00    0000  PP      PP  RR      RR  FF          LL          NNNN    NN  MM  MM  MM
          RRRRRRRR    MM      MM  00  00  00  PPPPPPPP    RRRRRRRR    FFFFFFFF    LL          NN  NN  NN  MM      MM
          RRRRRRRR    MM      MM  00  00  00  PPPPPPPP    RRRRRRRR    FFFFFFFF    LL          NN  NN  NN  MM      MM
          RR  RR      MM      MM  0000    00  PP          RR  RR      FF          LL          NN    NNNN  MM      MM
          RR  RR      MM      MM  0000    00  PP          RR  RR      FF          LL          NN    NNNN  MM      MM
          RR    RR    MM      MM  00      00  PP          RR    RR    FF          LL          NN      NN  MM      MM      ....
          RR    RR    MM      MM  00      00  PP          RR    RR    FF          LL          NN      NN  MM      MM      ....
          RR      RR  MM      MM    000000    PP          RR      RR  FF          LLLLLLLLLL  NN      NN  MM      MM      ....
          RR      RR  MM      MM    000000    PP          RR      RR  FF          LLLLLLLLLL  NN      NN  MM      MM      ....


          LL            IIIIII      SSSSSSSS
          LL            IIIIII      SSSSSSSS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II        SSSSSS
          LL              II        SSSSSS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LLLLLLLLLL    IIIIII    SSSSSSSS
          LLLLLLLLLL    IIIIII    SSSSSSSS
```

RMO
V04

Table of contents

```
    (3)       221         DECLARATIONS
    (4)       262         RM$PRFLNM, Filename Processing Routine
    (5)       413         RM$ASSIGN, Assign a Channel
    (6)       585         GETDEV_CHAR - Determine the device characteristics
    (15)     1070         RM$RET_DEV_CHAR - Set device characteristics into FAB
```

```
0000            .IF     DF,STASWITCH
0000            $BEGIN  STAPRFLNM,000,RM$RMS0,<PROCESS FILE NAME>
0000            .IFF    ;STASWITCH
0000            $BEGIN  RMOPRFLNM,033,RM$RMS0,<PROCESS FILE NAME>
0000            .ENDC   ;STASWITCH
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
```

```
0000    ;++
0000    ; Facility: RMS32
0000    ;
0000    ; Abstract:
0000    ;               This routine performs RMS32 file name processing.
0000    ;
0000    ; Environment:
0000    ;               Star processor running Starlet exec.
0000    ;
0000    ; Author: L. F. Laverdure         Creation Date: 4-JAN-1977
0000    ;
0000    ; Modified By:
0000    ;
0000    ;       V03-033 RAS0329         Ron Schaefer                    31-Jul-1984
0000    ;               Fix RM$PRFLNM to set GET access internally if EXE access is
0000    ;               requested; this is needed for execute-only command procedures.
0000    ;
0000    ;       V03-032 RAS0296         Ron Schaefer                    18-Apr-1984
0000    ;               Add secondary device name item code to $GETDVI.
0000    ;               This name is returned in NAM$T_DVI if the device is spooled.
0000    ;
0000    ;       V03-031 RAS0289         Ron Schaefer                    6-Apr-1984
0000    ;               Add notranslation flag to call to $ASSIGN; change
0000    ;               to WLD error when wildcard detected.
0000    ;
0000    ;       V03-030 RAS0268         Ron Schaefer                    12-Mar-1984
0000    ;               Move searchlist loop decision to caller of RM$PRFLNM[ALT].
0000    ;
0000    ;       V03-029 DGB0026         Donald G. Blair                 11-Mar-1984
0000    ;               Change BSBW to JSB to fix broken branch.
0000    ;
0000    ;       V03-028 RAS0242         Ron Schaefer                    23-Jan-1984
0000    ;               Fix bugchecks caused by lack of a fwa (valid R10)
0000    ;               on calls to RM$PRFLNM or error returns from RM$XPFN.
0000    ;
0000    ;       V03-027 RAS0229         Ron Schaefer                    5-Jan-1984
0000    ;               Return an error if $OPEN/$CREATE specifies a directory
0000    ;               containing "..." after expansion.   Related file
0000    ;               processing will resolve ellipses normally.
0000    ;
0000    ;       V03-026 RAS0225         Ron Schaefer                    20-Dec-1983
0000    ;               Fix stream PPF file with FTN carriage control problem;
0000    ;               eliminate setting of NAM$V_CNCL_DEV and NAM$V_ROOT_DIR
0000    ;               in RM$ASSIGN as this is now done by RM$EXPSTRING.
0000    ;
0000    ;       V03-025 RAS0212         Ron Schaefer                    15-Nov-1983
0000    ;               Fix version/implementation skew in RM$ASSIGN by
0000    ;               conditionalizing this module for both RMS and
0000    ;               stand-alone BACKUP on the STASWITCH parameter.
0000    ;
0000    ;       V03-024 RAS0209         Ron Schaefer                    4-Nov-1983
0000    ;               Clean-up returned device characteristics by defining
0000    ;               a routine RM$RET_DEV_CHAR to do a uniform job.
0000    ;
0000    ;       V03-023 RAS0198         Ron Schaefer                    6-Oct-1983
0000    ;               Change RM$PRFLNM[ALT] logic to not have an input parameter.
0000    ;
```

```
0000  ;       V03-022 SHZ0001         Stephen H. Zalewski        12-Sep-1983
0000  ;               Add a new itemcode to the itemcode list for the $GETDVI.
0000  ;
0000  ;       V03-021 RAS0183         Ron Schaefer               2-Sep-1983
0000  ;               Make searchlists be non-wildcard characters.
0000  ;
0000  ;       V03-020 KBT0560         Keith B. Thompson          21-Jul-1983
0000  ;               Set name block flags correctly
0000  ;
0000  ;       V03-019 SHZ0001         Stephen H. Zalewski        26-Jun-1983
0000  ;               Get full device name (node$device_name) from GETDVI.
0000  ;
0000  ;       V03-018 KBT0538         Keith B. Thompson          6-Jun-1983
0000  ;               Turn on search list
0000  ;
0000  ;       V03-017 KBT0515         Keith B. Thompson          23-May-1983
0000  ;               RM$XPFN moved
0000  ;
0000  ;       V03-016 KBT0510         Keith B. Thompson          5-May-1983
0000  ;               Use RM$DEALLOCATE_FWA add some search list stuff and
0000  ;               redo rooted directories
0000  ;
0000  ;       V03-015 RAS0146         Ron Schaefer               18-Apr-1983
0000  ;               Fix FWA$C_FIBLEN assume to have slightly more
0000  ;               tolerance for ACP FIB length changes.
0000  ;
0000  ;       V03-014 RAS0143         Ron Schaefer               11-Apr-1983
0000  ;               Fix device name buffer length to be 16, and change
0000  ;               $GETDVI to $GETDVIW.
0000  ;
0000  ;       V03-013 RAS0139         Ron Schaefer               24-Mar-1983
0000  ;               Undo RAS0129.  Not returning the true RAT and RFM values
0000  ;               is a "bug", since many programs do not set any values
0000  ;               on $OPEN and expect to find out what the RAT and RFM values
0000  ;               are from RMS.
0000  ;
0000  ;       V03-012 RAS0129         Ron Schaefer               4-Mar-1983
0000  ;               Do not destroy the user's RAT and RFM values for
0000  ;               indirect PPFs.
0000  ;
0000  ;       V03-011 JWH0187         Jeffrey W. Horn            15-Feb-1983
0000  ;               Add volume lable to list of items retrieved with
0000  ;               $GETDVI.
0000  ;
0000  ;       V03-010 KBT0485         Keith B. Thompson          4-Feb-1983
0000  ;               Deallocate all pages off the ifab free list
0000  ;
0000  ;       V03-009 KBT0473         Keith B. Thompson          24-Jan-1983
0000  ;               Fix kbt0469, provide the correct size to RM$RETSPC.
0000  ;
0000  ;       V03-008 KBT0469         Keith B. Thompson          24-Jan-1983
0000  ;               Deallocate the fwa on indirect ppf open
0000  ;
0000  ;       V03-007 KBT0432         Keith B. Thompson          3-Dec-1982
0000  ;               Change the way the device name is stored in shrfilbuf
0000  ;
0000  ;       V03-006 KBT0407         Keith B. Thompson          10-Nov-1982
```

```
0000  ;               Fix a broken assume (from unfolding fwa) and call new
0000  ;               $getdvi system service
0000  ;
0000  ;       V03-005 KBT0212         Keith B. Thompson       23-Aug-1982
0000  ;               Reorganize psects
0000  ;
0000  ;       V03-004 KBT0098         Keith B. Thompson       13-Jul-1982
0000  ;               Clean up psects
0000  ;
0000  ;       V03-003 KEK0026         K. E. Kinnear           23-Mar-1982
0000  ;               More work like KEK0022 -- correctly allow foreign devices
0000  ;               to have ANSI-"a" names. Also add back in NFS to FOREIGN
0000  ;               branch deleted by RAS0067.
0000  ;
0000  ;       V03-002 RAS0079         Ron Schaefer            17-Mar-1982
0000  ;               Add translation table mask support (FAB$B_DSBMSK).
0000  ;
0000  ;       V03-001 RAS0080         Ron Schaefer            17-Mar-1982
0000  ;               Correct stream format carriage control for indirect PPFs,
0000  ;               both terminals and files.
0000  ;
0000  ;       V02-044 RAS0067         Ron Schaefer            9-Feb-1982
0000  ;               Re-arrange execution paths so that FWA$T_SHRFILDEV
0000  ;               is filled in for all devices.  This field is used to
0000  ;               return a canonical device name in NAM$T_DVI.
0000  ;
0000  ;       V02-043 KEK0022         K. E. Kinnear           2-Feb-1982
0000  ;               Open restrictions on quoted strings to all non-disk
0000  ;               (actually non-DIR) devices.
0000  ;
0000  ;       V02-042 KEK0018         K. E. Kinnear           18-Jan-1982
0000  ;               Modifications to allow ANSI-"a" filespecs only on
0000  ;               magtape devices.  Also, remove all references to
0000  ;               NAM$x_QUOTED and replace with NAM$x_NAME.
0000  ;
0000  ;       V02-041 RAS0060         Ron Schaefer            15-Jan-1982
0000  ;               Fix temp buffer usage for concealed devices;
0000  ;               force stream PPF terminals to have RAT=CR.
0000  ;
0000  ;       V02-040 RAS0058         Ron Schaefer            8-Jan-1982
0000  ;               Correct concealed device error path so as to return
0000  ;               $ASSIGN errors; and fix bad code saving FWA$T_SHRFILDEV.
0000  ;
0000  ;       V02-039 RAS0047         Ron Schaefer            18-Nov-1981
0000  ;               Fix the UIC form of directories under rooted directories,
0000  ;               in order to make the AME and compatibility mode work.
0000  ;
0000  ;       V02-038 KPL0002         Peter Lieberwirth       15-Nov-1981
0000  ;               Fix RAS0045 to not use FWA$T_XLTBUFF1 in RM$ASSIGN.
0000  ;               Use unnamed space at end of first page instead.
0000  ;               (Fix this better ASAP!)
0000  ;
0000  ;       V02-037 RAS0045         Ron Schaefer            11-Nov-1981
0000  ;               Complete KPL0001 by changing the reference from
0000  ;               FWA$T_WILD to FWA$T_SHRFILDEV.  Also fix bad register
0000  ;               reference.
0000  ;
```

```
0000  ;       V02-036 RAS0040         Ron Schaefer            16-Oct-1981
0000  ;               Implement rooted directories for concealed devices.
0000  ;               Restructure RM$ASSIGN to parse the translation of the
0000  ;               concealed device name, assign a channel to the concealed
0000  ;               device and set the internal MFD to the DID of the
0000  ;               root directory.
0000  ;
0000  ;       V02-035 KPL0001         P Lieberwirth           7-Oct-1981
0000  ;               Save unit number and device name in FWA$T_WILD field for
0000  ;               later use in naming shared file.
0000  ;
0000  ;       V02-034 RAS0025         Ron Schaefer            18-Aug-1981
0000  ;               Allow $GET/$PUT for UDF seq org files, except as
0000  ;               indirect PPF.
0000  ;
0000  ;--
0000
```

```
                 0000          .SBTTL  DECLARATIONS
                 0000
                 0000   ;
                 0000   ; Include Files:
                 0000   ;
                 0000
                 0000   ;
                 0000   ; Macros:
                 0000   ;
                 0000
                 0000          $ASSIGNDEF
                 0000          $DEVDEF
                 0000          $DVIDEF
                 0000          $FABDEF
                 0000          $FIBDEF
                 0000          $FSCBDEF
                 0000          $FWADEF
                 0000          $IFBDEF
                 0000          $IMPDEF
                 0000          $NAMDEF
                 0000          $PSLDEF
                 0000          $RMSDEF
                 0000
                 0000          .IF     DF,STASWITCH
                 0000          $CCBDEF
                 0000          $DCDEF
                 0000          $UCBDEF
                 0000          .ENDC   ;STASWITCH
                 0000
                 0000   ;
                 0000   ; Equated Symbols:
                 0000   ;
                 0000
        00000020 0000          FOP=FAB$L_FOP*8                  ; bit offset to fop
                 0000
                 0000   ;
                 0000   ; Own Storage:
                 0000   ;
                 0000
```

```
0000          .IF     NDF,STASWITCH
0000          .SBTTL  RM$PRFLNM, Filename Processing Routine
0000
0000    ;++
0000    ;
0000    ; RM$PRFLNM -- Filename Processing Routine.
0000    ;
0000    ;  This routine first sets up various file access and sharing bits,
0000    ;  then allocates a buffer and bdb to use as a scratch work area for building
0000    ;  the expanded filename string and interfacing with f11acp, calls rm$xpfn
0000    ;  to expand the filename string, assigns an i/o channel, and finally
0000    ;  retrieves the device characteristics, filling in the associated
0000    ;  IFAB and FAB fields.
0000    ;
0000    ;  If the result of the file name expansion indicates that the file in
0000    ;  question is an indirectly accessed process permanent file, no i/o
0000    ;  channel need be assigned, as this has already been done.  Instead,
0000    ;  an ifi is constructed that points this fab at the associated process
0000    ;  permanent file.  This has the side effect of turning a $create call for
0000    ;  an indirectly accessed process permanent file into an $open.  This is
0000    ;  done by returning to the $OPEN code rather than the $CREATE code.
0000    ;
0000    ;  If an error occurs, cleanup may be required to
0000    ;  deallocate the bdb and buffer and deassign the channel.
0000    ;
0000    ; Calling Sequence:
0000    ;
0000    ;       BSBW    RM$PRFLNM
0000    ;
0000    ; Input Parameters:
0000    ;
0000    ;       R11     impure area address
0000    ;       R10     IFAB address
0000    ;       R9      IFAB address
0000    ;       R8      FAB address
0000    ;       R0      input error status (only if search list operation)
0000    ;
0000    ; Implicit Inputs:
0000    ;
0000    ;       The contents of the various FAB and IFAB fields
0000    ;       (see functional spec for details), in particular,
0000    ;       the various file name specification fields.
0000    ;
0000    ; Output Parameters:
0000    ;
0000    ;       R10     FWA address
0000    ;       R0      status code (could be R0 input status)
0000    ;       R1 thru R5 destroyed
0000    ;       none
0000    ;
0000    ; Implicit Outputs:
0000    ;
0000    ;       Various fab, fwa, and ifab fields are filled in (see
0000    ;       functional spec for details).
0000    ;       FWA$Q_FIB initialized.
0000    ;       device fields in the ifab filled in.
0000    ;
```

```
                             0000   318 ; Completion Codes:
                             0000   319 ;
                             0000   320 ;       Standard rms, in particular, success, dev,
                             0000   321 ;       chn, and dme, in addition to the codes returned
                             0000   322 ;       by RM$XPFN.
                             0000   323 ;
                             0000   324 ; Side Effects:
                             0000   325 ;
                             0000   326 ;       See note above on change of $create into $open for indirect ppf.
                             0000   327 ;
                             0000   328 ;--
                             0000   329
                             0000   330 RM$PRFLNM::
                             0000   331
                             0000   332 ;
                             0000   333 ;  Fill in fac IFAB field from fab handling defaults and setting summary
                             0000   334 ;  write access bit as required.
                             0000   335 ;
                             0000   336
        22 A9   16 A8   90   0000   337             MOVB    FAB$B_FAC(R8),IFB$B_FAC(R9)
                   0B   13   0005   338             BEQL    SETGET                          ; branch if null
     06 22 A9      07   E0   0007   339             BBS     #FAB$V_EXE,IFB$B_FAC(R9),SETGET ; or if EXE desired
                             000C   340
                             000C   341 ;
                             000C   342 ;  Entry point for RMSOCREATE (IFB$B_FAC already set)
                             000C   343 ;
                             000C   344
                             000C   345 RM$PRFLNMALT::
                   1C   93   000C   346             BITB    #FAB$M_UPD!FAB$M_DEL!FAB$M_TRN,-
                22 A9        000E   347                     IFB$B_FAC(R9)
                   04   13   0010   348             BEQL    SETWRT                          ; branch if not upd, del, or trn
                             0012   349                                                     ; accessed, else imply get access.
        22 A9      02   88   0012   350 SETGET: BISB2   #FAB$M_GET,IFB$B_FAC(R9)            ; default to get access
        22 A9      1D   93   0016   351 SETWRT: BITB    #FAB$M_PUT!FAB$M_UPD!FAB$M_DEL!FAB$M_TRN,IFB$B_FAC(R9)
                   04   13   001A   352             BEQL    20$                             ; branch if none of the varieties
                             001C   353                                                     ; of write access is specified.
                             001C   354             SSB     #IFB$V_WRTACC,(R9)              ; set summary bit
                             0020   355
                             0020   356 ;
                             0020   357 ;  Go expand file name.
                             0020   358 ;
                             0020   359
     00000000'EF   16        0020   360 20$:        JSB     RM$XPFN                         ; get fully qualified file name
             3B 50   E9      0026   361             BLBC    R0,60$                          ; quit on error
                             0029   362
                             0029   363 ;
                             0029   364 ; Check for residual wild characters in file name.
                             0029   365 ; If FWA$V_QUOTED, then skip check.  This can either be ANSI-"a" filespecs
                             0029   366 ; where FWA$Q_NAME really holds a quoted string, or it could be a network
                             0029   367 ; quoted string, where there is nothing in FWA$Q_NAME.
                             0029   368 ;
                             0029   369
        30 6A      18   E1   0029   370             BBC     #FWA$V_WILDCARD,(R10),50$       ; don't check if no wilds
        14 6A      1A   E0   002D   371             BBS     #FWA$V_QUOTED,(R10),40$         ; don't check if quoted
      0170 CA      2A   3A   0031   372             LOCC    #^A'*',FWA$Q_NAME(R10),-        ; any *'s in name?
           0174 DA           0036   373                     @FWA$Q_NAME+4(R10)
                   2D   12   0039   374             BNEQ    90$                             ; if neq yes
```

```
0170 CA    25   3A  003B   375          LOCC    #^A'%',FWA$Q_NAME(R10),-           ; any %'s in name
     0174 DA        0040   376                  @FWA$Q_NAME+4(R10)
           23   12  0043   377          BNEQ    90$                                ; if neq yes
                    0045   378
                    0045   379  ;
                    0045   380  ; Check for residual ellipses in the directory spec.  Related file
                    0045   381  ; processing should resolve valid uses previously.  Unfortunately,
                    0045   382  ; no simple check is possible -- one has to look at each directory
                    0045   383  ; spec for the FSCB$V_ELIPS bit.
                    0045   384  ;
                    0045   385
   14 6A   1C   E1  0045   386 40$:     BBC     #FWA$V_WILD_DIR,(R10),50$          ; don't bother if no wild dirs
      03   1D   EF  0049   387          EXTZV   #FWA$V_DIR_LVLS,#FWA$S_DIR_LVLS,-; get # of dirs to check
      50   6A       004C   388                  (R10),R0                           ;
51  0130   CA   7E  004E   389          MOVAQ   FWA$Q_DIR1(R10),R1                 ; ptr to dir descriptors
   17 61   10   E0  0053   390 45$:     BBS     #FSCB$V_ELIPS,(R1),100$            ; ... is a no-no
      51   08   C0  0057   391          ADDL2   #8,R1                              ; next directory
      F6   50   F4  005A   392          SOBGEQ  R0,45$                             ; while they last
                    005D   393
                    005D   394  ;
                    005D   395  ;  Check for indirect open of process permanent file.
                    005D   396  ;
                    005D   397
   0C AA   95       005D   398 50$:     TSTB    FWA$B_ESCFLG(R10)                  ; did we get a ppf flag?
      03   12       0060   399          BNEQ    80$                                ; branch if so
      10   10       0062   400          BSBB    RM$ASSIGN                          ; assign a channel
           05       0064   401 60$:     RSB                                        ; exit
                    0065   402
    01E3   31       0065   403 80$:     BRW     INDPPF                             ; process indirect ppf file
                    0068   404
                    0068   405 90$:     RMSERR  WLD                                ; give wildcard file name error
           05       006D   406          RSB
                    006E   407
                    006E   408 100$:    RMSERR  DIR                                ; give bad directory error
           05       0073   409          RSB
                    0074   410
                    0074   411          .ENDC   ;STASWITCH
```

```
                      0074          .SBTTL  RM$ASSIGN, Assign a Channel
                      0074  ;++
                      0074  ;
                      0074  ; RM$ASSIGN
                      0074  ;
                      0074  ;  Now assign a channel and get the associated device characteristics.
                      0074  ;
                      0074  ;  Assign in MAX( requested mode,caller's mode ) if NFS or UFO set,
                      0074  ;  else assign in exec mode.
                      0074  ;
                      0074  ;--
                      0074
                      0074  RM$ASSIGN::
                      0074
                      0074          .IF     NDF,STASWITCH
                      0074
 4E 6A    19     E0   0074          BBS     #FWA$V_NODE,(R10),NTASGN        ; branch if network request
                      0078
                      0078          .ENDC   ;STASWITCH
                      0078
                      0078
                      0078  ;
                      0078  ; Note: Device name already has been prefixed with an underscore, so
                      0078  ;       assign system service will not attempt to translate it again.
                      0078  ;
                      0078  ASSIGN:
          00     DD   0078          PUSHL   #0                              ; no mailbox
                      007A
                      007A          ASSUME  ASSIGN$_MBXNAM  EQ      ASSIGN$_ACMODE+4
                      007A
          01     DD   007A          PUSHL   #PSL$C_EXEC                     ; exec mode
                      007C
                      007C          ASSUME  FAB$V_UFO       GE      16
                      007C          ASSUME  FAB$V_NFS       GE      16
                      007C
 06 A8    03     93   007C          BITB    #<FAB$M_UFO!FAB$M_NFS>@-16,FAB$L_FOP+2(R8)
          10     13   0080          BEQL    40$                             ; branch if neither ufo nor nfs set
          02     EF   0082          EXTZV   #FAB$V_CHAN_MODE,-              ; replace with requested mode
          02          0084                  #FAB$S_CHAN_MODE,-              ;
 6E    4A A8          0085                  FAB$B_ACMODES(R8),(SP)          ;
 6E    0A A9     91   0088          CMPB    IFB$B_MODE(R9),(SP)             ; compare with caller's mode
          04     1B   008C          BLEQU   40$                             ; maximize
 6E    0A A9     9A   008E          MOVZBL  IFB$B_MODE(R9),(SP)             ; switch to caller's mode
                      0092
                      0092          ASSUME  ASSIGN$_CHAN    EQ      ASSIGN$_ACMODE-4
                      0092
                      0092  40$:    ssb     #7,(sp)                         ; set notranslate flag
       20 A9     DF   0096          PUSHAL  IFB$W_CHNL(R9)                  ; get channel # back here
                      0099
                      0099          ASSUME  ASSIGN$_DEVNAM  EQ      ASSIGN$_CHAN-4
                      0099
     00E0 CA     7F   0099          PUSHAQ  FWA$Q_DEVICE(R10)               ; device name descriptor
          39     E1   009D          BBC     #FWA$V_CONCEAL_DEV,-            ; is this a concealed device
       05 6A          009F                  (R10),45$                       ;
     00E8 CA     7E   00A1          MOVAQ   FWA$Q_CONCEAL_DEV(R10),-; yes replace the descriptor
          6E          00A5                  (SP)                            ;
                      00A6
```

```
                            00A6   470          ASSUME  ASSIGN$_NARGS    EQ      4
                            00A6   471
        00000000'9F   04  FB  00A6   472 45$:     CALLS   #4,@#SYS$ASSIGN           ; and do the assign
                  27 50  E9  00AD   473          BLBC    R0,ERRASGN
                            00B0   474
                            00B0   475 ;
                            00B0   476 ;  Assign succeeded  -  get and set device characteristics.
                            00B0   477 ;
                            00B0   478
                  002C   30  00B0   479          BSBW    GETDEV_CHAR
                            00B3   480
                            00B3   481          .IF     NDF,STASWITCH
                            00B3   482
               0D 6B    E9  00B3   483          BLBC    (R11),DIRECTPPF1
                            00B6   484
                            00B6   485 ;
                            00B6   486 ; Set up fib descriptor.
                            00B6   487 ;
                            00B6   488
                            00B6   489 RM$SETFIB::
                            00B6   490
                            00B6   491          ASSUME  FIB$C_LENGTH    LE      FWA$C_FIBLEN
                            00B6   492
    10 AA     0040 8F   3C  00B6   493          MOVZWL  #FIB$C_LENGTH,FWA$Q_FIB(R10)     ; set length of fib
              01F4 CA   9E  00BC   494          MOVAB   FWA$T_FIBBUF(R10),-              ; set address of fib buffer
                 14 AA      00C0   495                  FWA$Q_FIB+4(R10)                 ;
                            00C2   496
                      05    00C2   497 ERROR:   RSB
                            00C3   498
                            00C3   499 DIRECTPPF1:
                  013B   31  00C3   500          BRW     DIRECTPPF                        ; branch if process-perm file
                            00C6   501
                            00C6   502          .ENDC   ;STASWITCH
                            00C6   503
                            00C6   504          .IF     DF,STASWITCH
                            00C6   505
                            00C6   506          BBS     S^#EXE$V_INIT,EXE$GL_FLAGS,110$ ; br if running online
                            00C6   507          CMPB    FWA$L_DEV_CLASS(R10),#DC$_DISK  ; disk device?
                            00C6   508          BEQL    100$                            ; br if disk
                            00C6   509          CMPB    FWA$L_DEV_CLASS(R10),#DC$_TAPE  ; tape device?
                            00C6   510          BNEQ    110$                            ; br if not tape
                            00C6   511 100$:    SSB     #DEV$V_FOR,IFB$L_PRIM_DEV(R9) ; simulate mount/foreign
                            00C6   512          SSB     #DEV$V_FOR,IFB$L_AS_DEV(R9) ; simulate mount/foreign
                            00C6   513          $CMKRNL_S       STA_VOL_VAL     ; set volume valid
                            00C6   514          BLBC    R0,ERRASGN              ; br if verifychan failed
                            00C6   515          $QIOW_S -
                            00C6   516                  FUNC = #IO$_PACKACK,-
                            00C6   517                  EFN = #IMP$C_ASYQIOEFN,-        ; throw-away efn
                            00C6   518                  CHAN = IFB$W_CHNL(R9)
                            00C6   519          BLBC    R0,ERRASGN              ; br if QIO failed
                            00C6   520 110$:    $DASSGN_S       CHAN=IFB$W_CHNL(R9)
                            00C6   521          CLRW    IFB$W_CHNL(R9)          ; clear channel number
                            00C6   522          RSB                             ; exit
                            00C6   523
                            00C6   524          .ENDC   ;STASWITCH
                            00C6   525
                            00C6   526          .IF     NDF,STASWITCH
```

```
                        00C6   527 ;++
                        00C6   528 ;
                        00C6   529 ;  Assign a channel to the network device and get and set its device
                        00C6   530 ;  characteristics.
                        00C6   531 ;
                        00C6   532 ;--
                        00C6   533
                        00C6   534 NTASGN:
         00000000'EF  16  00C6   535         JSB     NT$ASSIGN               ; assign a channel
               08 50  E9  00CC   536         BLBC    R0,ERRASGN              ; branch on error
                  0E  10  00CF   537         BSBB    GETDEV_CHAR             ; get network device characteristics
         00000000'EF  17  00D1   538         JMP     NT$MOD_DEV_CHAR         ; modify device characteristics
                        00D7   539                                         ;  and return to caller
                        00D7   540         .ENDC   ;STASWITCH
                        00D7   541
                        00D7   542 ;
                        00D7   543 ;  Handle assign error.
                        00D7   544 ;
                        00D7   545
                        00D7   546 ERRASGN:
                        00D7   547
                        00D7   548         .IF     NDF,STASWITCH
                        00D7   549
                        00D7   550         RMSERR  CHN,R1                  ; unspecified chnl assgn failure
               FF21'  31  00DC   551         BRW     RM$MAPERR               ; map error and return to caller
                        00DF   552
                        00DF   553         .ENDC   ;STASWITCH
                        00DF   554
                        00DF   555         .IF     DF,STASWITCH
                        00DF   556
                        00DF   557         MOVL    IFB$L_LAST_FAB(R9),R1   ; get FAB address
                        00DF   558         MOVL    R0,FAB$L_STV(R1)        ; set STV value
                        00DF   559         RMSERR  CHN,R0                  ; set R0
                        00DF   560         RSB                             ; exit
                        00DF   561
                        00DF   562         .ENDC   ;STASWITCH
                        00DF   563
                        00DF   564 ;
                        00DF   565 ; Kernel mode routine to set volume valid and mark device mounted
                        00DF   566 ; for use in stand-alone Backup.
                        00DF   567 ;
                        00DF   568
                        00DF   569         .IF     DF,STASWITCH
                        00DF   570
                        00DF   571 STA_VOL_VAL:
                        00DF   572         .WORD   ^M<R2,R3,R4,R5>         ; kernel mode routine to set valid
                        00DF   573         MOVZWL  IFB$W_CHNL(R9),R0       ; load channel number
                        00DF   574         JSB     @#IOC$VERIFYCHAN        ; get CCB address for channel
                        00DF   575         BLBC    R0,210$                 ; br if failed
                        00DF   576         MOVL    CCB$L_UCB(R1),R1        ; get UCB address from CCB
                        00DF   577         BISW2   #UCB$M_VALID,UCB$W_STS(R1)      ; set software valid bit
                        00DF   578         BISL2   #DEV$M_MNT!DEV$M_FOR,UCB$L_DEVCHAR(R1)
                        00DF   579                                         ; set mounted foreign bits
                        00DF   580         MOVL    #1,R0                   ; indicate success
                        00DF   581 210$:   RET                             ; return
                        00DF   582
                        00DF   583         .ENDC   ;STASWITCH
```

```
                    00DF          .SBTTL  GETDEV_CHAR - Determine the device characteristics
                    00DF  ;
                    00DF  ; Subroutine to get and set device characteristics.
                    00DF  ;
                    00DF  ;       R0-R2   destroyed
                    00DF  ;
                    00DF  ;
                    00DF  ; The device characteristics wanted are:
                    00DF  ;
                    00DF  ;       DVI$_DEVCHAR                =>      IFB$L_PRIM_DEV
                    00DF  ;       DVI$_DEVCHAR (sec)          =>      IFB$L_AS_DEV
                    00DF  ;       DVI$_DEVBUFSIZ              =>      IFB$L_DEVBUFSIZ
                    00DF  ;       DVI$_DEVBUFSIZ (sec)        =>      IFB$L_ASDEVBSIZ
                    00DF  ;       DVI$_DEVCLASS               =>      FWA$L_DEVCLASS
                    00DF  ;       DVI$_FULLDEVNAM             =>      FWA$T_SHRFILBUF (pointed to by shrfil)
                    00DF  ;       DVI$_FULLDEVNAM (sec)       =>      FWA$T_AS_SHRFILBUF (pointed to by as_shrfil)
                    00DF  ;       DVI$_DEVLOCKNAM             =>      FWA$T_SHRFIL_LCKNAM (pointed to by shrfil_lc
                    00DF  ;       DVI$_RECSIZ                 =>      FWA$L_RECSIZ
                    00DF  ;
                    00DF
                    00DF  GETDEV_CHAR:
                    00DF
                    00DF  ;
                    00DF  ; Create item list on stack
                    00DF  ;
      52   5E   D0  00DF          MOVL    SP,R2                   ; save addr to calculate size
           00   DD  00E2          PUSHL   #00                     ; mark end of list
                    00E4
                    00E4  ;
                    00E4  ; Create item for devchar value
                    00E4  ;
                    00E4
           00   DD  00E4          PUSHL   #00                     ; return size
           69   DF  00E6          PUSHAL  IFB$L_PRIM_DEV(R9)      ; return buffer
  00020004 8F   DD  00E8          PUSHL   #<DVI$_DEVCHAR@16>+4    ; item code and buffer size
                    00EE
                    00EE  ;
                    00EE  ; Create item for secondary devchar value
                    00EE  ;
                    00EE
           00   DD  00EE          PUSHL   #00                     ; return size
      008C C9   DF  00F0          PUSHAL  IFB$L_AS_DEV(R9)        ; return buffer
  00030004 8F   DD  00F4          PUSHL   #<<DVI$_DEVCHAR!DVI$C_SECONDARY>@16>+4  ; item code and buffer size
                    00FA
                    00FA  ;
                    00FA  ; Create item for devbufsiz value
                    00FA  ;
                    00FA
           00   DD  00FA          PUSHL   #00                     ; return size
        48 A9   DF  00FC          PUSHAL  IFB$L_DEVBUFSIZ(R9)     ; return buffer
  00080004 8F   DD  00FF          PUSHL   #<DVI$_DEVBUFSIZ@16>+4  ; item code and buffer size
                    0105
                    0105  ;
                    0105  ; Create item for secondary devbufsiz value
                    0105  ;
                    0105
           00   DD  0105          PUSHL   #00                     ; return size
      0094 C9   DF  0107           PUSHAL  IFB$L_ASDEVBSIZ(R9)       ; return buffer
  00090004 8F   DD  010B           PUSHL   #<<DVI$_DEVBUFSIZ!DVI$C_SECONDARY>@16>+4; item code and buffer size
                    0111
                    0111     ;
                    0111     ; Create item for devclass value
                    0111     ;
                    0111
        00      DD  0111           PUSHL   #00                       ; return size
        1C AA   DF  0113           PUSHAL  FWA$L_DEV_CLASS(R10)      ; return buffer
  00040004 8F   DD  0116           PUSHL   #<DVI$_DEVCLASS@16>+4     ; item code and buffer size
                    011C
                    011C     ;
                    011C     ; Create item for devnam string
                    011C     ;
                    011C
      0190 CA   3F  011C           PUSHAW  FWA$Q_SHRFIL(R10)         ; return size
      0194 CA   DD  0120           PUSHL   FWA$Q_SHRFIL+4(R10)       ; return buffer
  00E80010 8F   DD  0124           PUSHL   #<DVI$_FULLDEVNAM@16>-
                    012A                   +FWA$S_SHRFILBUF          ; item code and buffer size
                    012A
                    012A     ;
                    012A     ; Create item for devlocknam string
                    012A     ;
                    012A
      0198 CA   3F  012A           PUSHAW  FWA$Q_SHRFIL_LCK(R10)     ; return size
      019C CA   DD  012E           PUSHL   FWA$Q_SHRFIL_LCK+4(R10)   ; return buffer
  00F00010 8F   DD  0132           PUSHL   #<DVI$_DEVLOCKNAM@16>-
                    0138                   +FWA$S_SHRFIL_LCKNAM      ; item code and buffer size
                    0138
                    0138     ;
                    0138     ; Create item for secondary devnam string
                    0138     ;
                    0138
      01A0 CA   3F  0138           PUSHAW  FWA$Q_AS_SHRFIL(R10)      ; return size
      01A4 CA   DD  013C           PUSHL   FWA$Q_AS_SHRFIL+4(R10)    ; return buffer
  00E90010 8F   DD  0140           PUSHL   #<<DVI$_FULLDEVNAM!DVI$C_SECONDARY>@16>-
                    0146                   +FWA$S_AS_SHRFILBUF       ; item code and buffer size
                    0146
                    0146     ;
                    0146     ; Create item for recsiz value
                    0146     ;
                    0146
        00      DD  0146           PUSHL   #00                       ; return size
        20 AA   DF  0148           PUSHAL  FWA$L_RECSIZ(R10)         ; return buffer
  00180004 8F   DD  014B           PUSHL   #<DVI$_RECSIZ@16>+4       ; item code and buffer size
                    0151
                    0151     ;
                    0151     ; Create item for volume label
                    0151     ;
                    0151
        00      DD  0151           PUSHL   #00                       ; return size
      0920 CA   DF  0153           PUSHAL  FWA$T_VOLNAM(R10)         ; return buffer
  0022000C 8F   DD  0157           PUSHL   #<DVI$_VOLNAM@16>+12      ; item code and buffer size
        51 5E   D0  015D           MOVL    SP,R1                     ; save addr for call
        5E 08   C2  0160           SUBL2   #8,SP                     ; make iosb
        50 5E   D0  0163           MOVL    SP,R0                     ; save addr for iosb
                    0166           $GETDVIW_S -
```

```
                           0166   699                    CHAN = IFB$W_CHNL(R9),- ; I/O channel number
                           0166   700                    EFN = #IMP$C_ASYQIOEFN,-; throw-away efn
                           0166   701                    IOSB = (R0),-           ; iosb for synchronizing
                           0166   702                    ITMLST = (R1)           ; item list
                           017D   703
           5E    52   D0   017D   704            MOVL    R2,SP                   ; restore stack
              7B 50   E9   0180   705            BLBC    R0,ERRASGN1             ; exit on error
                           0183   706
                           0183   707    ;
                           0183   708    ; Subtract one from the device name sizes in order to get rid of the ":"
                           0183   709    ; at the end of the name
                           0183   710    ;
         0190 CA      B7   0183   711            DECW    FWA$Q_SHRFIL(R10)
         01A0 CA      B7   0187   712            DECW    FWA$Q_AS_SHRFIL(R10)
                           018B   713
                           018B   714            .IF     NDF,STASWITCH
                           018B   715
                           018B   716    ;
                           018B   717    ; Deal with FOREIGN devices in a separate code section.  They all allow ANSI-"a"
                           018B   718    ; names, so don't worry about checking them.
                           018B   719    ;
                           018B   720
     1B 69    18      E0   018B   721            BBS     #DEV$V_FOR,IFB$L_PRIM_DEV(R9),FOREIGN ; branch if dev mntd foreign
     17 68    30      E0   018F   722            BBS     #FAB$V_NFS+FOP,(R8),FOREIGN     ; branch if non-file-struct.
     0C 69    05      E0   0193   723            BBS     #DEV$V_SQD,IFB$L_PRIM_DEV(R9),10$; branch if magtape
     08 69    03      E1   0197   724            BBC     #DEV$V_DIR,IFB$L_PRIM_DEV(R9),10$; branch if not dir device
                           019B   725
                           019B   726    ;
                           019B   727    ; At this point we are sure that we are not using an ANSI magtape device.
                           019B   728    ; Check to see if we have an ANSI quoted string in the FWA, denoted by
                           019B   729    ; FWA$V_QUOTED set and FWA$V_NODE not set.
                           019B   730    ;
                           019B   731
     04 6A    1A      E1   019B   732            BBC     #FWA$V_QUOTED,(R10),10$ ; quoted not set
     01 6A    19      E1   019F   733            BBC     #FWA$V_NODE,(R10),20$   ; networking, no problem
                           01A3   734
                           01A3   735            .ENDC   ;STASWITCH
                           01A3   736
                      05   01A3   737    10$:    RSB                             ; exit
                           01A4   738
                           01A4   739    20$:    RMSERR  FNM                     ; error in file name
                      05   01A9   740            RSB
                           01AA   741
```

```
                         01AA
                         01AA              .IF     NDF,STASWITCH
                         01AA      ;
                         01AA      ; User has requested non-file-structured operations (via the nfs fop bit)
                         01AA      ; or device was mounted foreign.
                         01AA      ;
                         01AA      ; Clear the 'dir' and 'sdi' device characteristics (directory, single directory)
                         01AA      ; and set the 'for' device characteristic (foreign) and set up various other
                         01AA      ; file attributes.
                         01AA      ;
                         01AA
                         01AA      FOREIGN:
          69    18   8A  01AA              BICB2   #DEV$M_DIR!DEV$M_SDI,IFB$L_PRIM_DEV(R9); say no directory
                         01AD              SSB     #DEV$V_FOR,IFB$L_PRIM_DEV(R9); say foreign device
       1F 69    05   E1  01B1              BBC     #DEV$V_SQD,IFB$L_PRIM_DEV(R9),CHKRND; branch if not magtape
       51    20 AA   3C  01B5              MOVZWL  FWA$L_RECSIZ(R10),R1       ; fixed recsize from mount
                35   13  01B9              BEQL    SETREC                     ; branch if not speced
       0C 69    32   E1  01BB              BBC     #IFB$V_CREATE,(R9),10$     ; branch if doing open
                         01BF
                         01BF      ;
                         01BF      ;  This is create.  Only allow blocking if rfm=fix and mrs = recordsize.
                         01BF      ;
                         01BF
       01    1F A8   91  01BF              CMPB    FAB$B_RFM(R8),#FAB$C_FIX; rfm = fix?
                2B   12  01C3              BNEQ    SETREC                     ; branch if not
       51    36 A8   B1  01C5              CMPW    FAB$W_MRS(R8),R1           ; same size?
                25   12  01C9              BNEQ    SETREC                     ; branch if not
       60 A9    51   B0  01CB      10$:    MOVW    R1,IFB$W_MRS(R9)           ; set fixed rec size from
                         01CF                                                 ; mount parameter.
          69    01   8A  01CF              BICB2   #DEV$M_REC,IFB$L_PRIM_DEV(R9); clear 'unit rec' char.
                11   11  01D2              BRB     SETFIX                     ; go set fixed record length
       18 69    1C   E1  01D4      CHKRND: BBC     #DEV$V_RND,IFB$L_PRIM_DEV(R9),SETREC; branch if not disk
    60 A9    48 A9   B0  01D8              MOVW    IFB$L_DEVBUFSIZ(R9),IFB$W_MRS(R9); say fixed 512 records
       74 A9    02   CE  01DD              MNEGL   #2,IFB$L_EBK(R9)           ; say large eof blk
                         01E1                                                 ; (RM1CREATE increments this)
       70 A9    01   CE  01E1              MNEGL   #1,IFB$L_HBK(R9)           ; say large allocation
       50 A9    01   90  01E5      SETFIX: MOVB    #FAB$C_FIX,IFB$B_RFMORG(R9); say fixed records len.
    52 A9    60 A9   B0  01E9              MOVW    IFB$W_MRS(R9),IFB$W_LRL(R9); set fixed rec len.
                03   11  01EE              BRB     EXIT                       ; continue
          69    01   88  01F0      SETREC: BISB2   #DEV$M_REC,IFB$L_PRIM_DEV(R9); say mt is unit rec
    05 22 A9    05   E0  01F3      EXIT:   BBS     #IFB$V_BIO,IFB$B_FAC(R9),10$; branch if block i/o
                         01F8              SSB     #IFB$V_BRO,IFB$B_FAC(R9); allow block i/o functions
                     05  01FD      10$:    RSB
                         01FE
                         01FE              .ENDC   ;STASWITCH
                         01FE
                         01FE      ERRASGN1:
               FED6  31  01FE              BRW     ERRASGN
```

```
                              0201
                              0201          .IF     NDF,STASWITCH
                              0201  ;
                              0201  ; Process-Permanent File  -  check device characteristcs for terminal
                              0201  ; and if so reassign the channel in super mode.
                              0201  ;
                              0201  ; Also reset device characteristics to indicate get/put access
                              0201  ; via dev.
                              0201  ;
                              0201
                              0201  DIRECTPPF:
       26 69    02      E1    0201          BBC     #DEV$V_TRM,IFB$L_PRIM_DEV(R9),SETDEV
                              0205
                              0205  ;
                              0205  ; Process permanent file is a terminal  -  reassign channel in super mode
                              0205  ; retain the exec mode channel until we have a super mode channel,
                              0205  ; so that the virtual terminal ucb will not disappear.  This,
                              0205  ; unfortunately, has the property that the user can get a spurious
                              0205  ; no more channels error.
                              0205  ;
                              0205
       52    20 A9      B0    0205          MOVW    IFB$W_CHNL(R9),R2          ; save exec channel number
                              0209          $ASSIGN_S                  -
                              0209                  FWA$Q_DEVICE(R10),-
                              0209                  IFB$W_CHNL(R9),-
                              0209                  #PSL$C_SUPER
          53    50      D0    021B          MOVL    R0,R3                      ; preserve assign status code
                              021E          $DASSGN_S       R2                 ; return the exec mode channel
       D3 53            E9    0228          BLBC    R3,ERRASGN1
                              022B
                              022B  ;
                              022B  ; Assume no change in dev chars; hence no need to reprocess.
                              022B  ;
                              022B  ;  If FAB$V_GET not set in fac, clear DEV$V_IDV.
                              022B  ;  If FAB$V_PUT not set in fac, clear DEV$V_ODV.
                              022B  ;
                              022B
 0A 22 A9       01      E0    022B  SETDEV: BBS     #FAB$V_GET,IFB$B_FAC(R9),10$; branch if 'get' on
                              0230          CSB     #DEV$V_IDV,IFB$L_PRIM_DEV(R9)
                              0234          CSB     #DEV$V_IDV,IFB$L_AS_DEV(R9)
                              023A
                              023A          ASSUME  FAB$V_PUT          EQ      0
                              023A
    0A 22 A9            E8    023A  10$:    BLBS    IFB$B_FAC(R9),20$          ; branch if 'put' on
                              023E          CSB     #DEV$V_ODV,IFB$L_PRIM_DEV(R9)
                              0242          CSB     #DEV$V_ODV,IFB$L_AS_DEV(R9)
          FE6B          31    0248  20$:    BRW     RM$SETFIB                  ; return to main line
                              024B
                              024B          .ENDC   ;STASWITCH
```

```
                          024B
                          024B          .IF     NDF,STASWITCH
                          024B  ;
                          024B  ;  Handle indirect open of process permanent file.
                          024B  ;
                          024B  ;  Perform various checks to see if operation possible.
                          024B  ;
                          024B  ;  (Note: This routine has the side effect of turning a $CREATE
                          024B  ;         on an inidrect process permanent file into an $OPEN)
                          024B  ;
                          024B
                          024B  INDPPF:
                          024B          $TSTPT  INDPPF
                          0251
                          0251          ASSUME  FAB$V_UFO       GE      16
                          0251          ASSUME  FAB$V_NFS       GE      16
                          0251
                          0251  ;
                          0251  ; Check for various invalid options.
                          0251  ;
                          0251
        06 A8    03    93 0251          BITB    #<FAB$M_UFO!FAB$M_NFS>@-16,FAB$L_FOP+2(R8)
                 55    12 0255          BNEQ    ERRFOP                          ; branch if any specified
              0D AA    95 0257          TSTB    FWA$B_ESCTYP(R10)               ; escape type 0?
                 56    12 025A          BNEQ    ERRLNE                          ; branch if not (not legal)
02    0E AA   02 0E    ED 025C          CMPZV   #14,#2,FWA$W_ESCIFI(R10),#^B10; escape ifi indicate a ppf?
                 4E    12 0262          BNEQ    ERRLNE                          ; branch if not
                          0264
                          0264  ;
                          0264  ;  Get address of process permanent file IFAB and check it out.
                          0264  ;
                          0264
              57 59    D0 0264          MOVL    R9,R7                           ; save ifab addr
                          0267
                          0267          ASSUME  FAB$V_PPF_IND   GE      8
                          0267
      0F AA   40 8F    88 0267          BISB2   #FAB$M_PPF_IND@-8,FWA$W_ESCIFI+1(R10); get non-privileged ppf ifi
         59   0E AA    3C 026C          MOVZWL  FWA$W_ESCIFI(R10),R9            ; and move it to r9
              50 06    D0 0270          MOVL    #IMP$C_IFABTBL/4,R0             ; ifab table offset/4
                 5B    DD 0273          PUSHL   R11                             ; save impure area pointer
               FD88'   30 0275          BSBW    RM$GTIADR                       ; get ifab address
            0800 8F    BA 0278          POPR    #^M<R11>                        ; restore image i/o ptr (save cc's)
                 41    13 027C          BEQL    ERRIFI                          ; branch if no ppf ifab
         0B   08 A9    91 027E          CMPB    IFB$B_BID(R9),#IFB$C_BID; is it an ifab?
                 34    12 0282          BNEQ    ERRBG3                          ; really bad if not!
                          0284
                          0284  ;
                          0284  ;  Got ppf IFAB o.k.
                          0284  ;
                          0284  ;  Insure org is sequential and rfm is not UDF
                          0284  ;  (Note: User fac is ignored.  Each operation will be checked.)
                          0284  ;
                          0284
                          0284          ASSUME  FAB$C_SEQ       EQ      0
                          0284
              23 A9    95 0284          TSTB    IFB$B_ORGCASE(R9)               ; seq file org?
                 3D    12 0287          BNEQ    ERRORG                          ; branch if not (not supported)
```

```
                          0289   899
                          0289   900            ASSUME  FAB$C_UDF         EQ       0
                          0289   901
            50 A9     95  0289   902            TSTB    IFB$B_RFMORG(R9)             ; undefined format?
               3F     13  028C   903            BEQL    ERRRFM                       ; branch if so (not supported)
                          028E   904
                          028E   905  ;
                          028E   906  ;  Check for valid rat value: must be only one of cr, ftn, or prn.
                          028E   907  ;  If prn set in rat must also be set for real file.
                          028E   908  ;
                          028E   909  ;  Isolate carriage control bits
                          028E   910  ;
               51     D4  028E   911            CLRL    R1                           ; zero-extend the field
  51   1E A8   F8 8F  8B  0290   912            BICB3   #255\<FAB$M_CR!FAB$M_FTN!FAB$M_PRN>,FAB$B_RAT(R8),R1
          09 69   1B  E1  0296   913            BBC     #DEV$V_ODV,IFB$L_PRIM_DEV(R9),5$; branch if input only
          05 51   02  E1  029A   914            BBC     #FAB$V_PRN,R1,5$             ; branch if not prn
       31 51 A9   02  E1  029E   915            BBC     #FAB$V_PRN,IFB$B_RAT(R9),ERRRAT; branch if ppf not prn
                          02A3   916
                          02A3   917            ASSUME  FAB$V_FTN EQ 0
                          02A3   918            ASSUME  FAB$V_CR EQ FAB$V_FTN+1
                          02A3   919            ASSUME  FAB$V_PRN EQ FAB$V_CR+1
                          02A3   920
       2C 17'8F   51  E1  02A3   921 5$:        BBC     R1,I^#^B00010111,ERRRAT ; branch if more than 1 ccl attr
                          02A8   922
                          02A8   923  ;
                          02A8   924  ;  Check for ppf IFAB busy setting busy if not in use.
                          02A8   925  ;
                          02A8   926
          38 69   20  E3  02A8   927 10$:       BBCS    #IFB$V_BUSY,(R9),IND_PPF_OK; branch if not busy
                          02AC   928
                          02AC   929
                          02AC   930 ERRFOP:
                          02AC   931            RMSERR  FOP                          ; invalid fop bits for ppf
                      05  02B1   932            RSB
                          02B2   933
                          02B2   934 ERRLNE:
                          02B2   935            RMSERR  LNE                          ; invalid equivalence string
                      05  02B7   936            RSB
                          02B8   937
                          02B8   938 ERRBG3:    RMSTBUG FTL$_BADIFAB                 ; invalid ifab
                          02BF   939
                          02BF   940 ERRIFI:
                          02BF   941            RMSERR  IFI                          ; invalid equiv. string ifi
               13     11  02C4   942            BRB     RESTORE_IFAB
                          02C6   943
                          02C6   944 ERRORG:
                          02C6   945            RMSERR  ORG                          ; ppf not sequential org
               0C     11  02CB   946            BRB     RESTORE_IFAB
                          02CD   947
                          02CD   948 ERRRFM:
                          02CD   949            RMSERR  RFM                          ; ppf UDF format
               05     11  02D2   950            BRB     RESTORE_IFAB
                          02D4   951
                          02D4   952 ERRRAT:
                          02D4   953            RMSERR  RAT                          ; bad rat value
                          02D9   954 RESTORE_IFAB:
            59 57     D0  02D9   955            MOVL    R7,R9                        ; restore temp. ifab addr
```

```
                       05  02DC     956             RSB                                       ; go handle error
                           02DD     957
                           02DD     958
                           02DD     959 ERRNAM:
                  02   BA  02DD     960             POPR    #^M<R1>                           ; get ppf ifab addr
                           02DF     961             CSB     #IFB$V_BUSY,(R1)                  ; clear busy
                       05  02E3     962             RSB                                       ; go clean up
                           02E4     963
```

```
                         02E4
                         02E4  ;
                         02E4  ;  The indirect open of the process permanent file is permissable.
                         02E4  ;  switch to using the ppf IFAB for the remainder of this open.
                         02E4  ;
                         02E4  ;  Must perform the following:
                         02E4  ;
                         02E4  ;          1. Set ppf_image bit.
                         02E4  ;          2. Copy caller's mode and arg list address to the ppf ifab.
                         02E4  ;             (Note: there will be no asb because no stall can have occurred.)
                         02E4  ;          3. Return the temporary ifab and fwa.
                         02E4  ;          4. Zero the ifi table entry to temporary ifab.
                         02E4  ;          5. Move address of pio impure area to r11 and copy saved sp
                         02E4  ;             to pio impure area.
                         02E4  ;
                         02E4
                         02E4  IND_PPF_OK:
                         02E4          SSB     #IFB$V_PPF_IMAGE,(R9)    ; indicate indirect processing
                         02E8                                           ;  of ppf
    0A A9   0A A7   90   02E8          MOVB    IFB$B_MODE(R7),IFB$B_MODE(R9); save caller's mode
    18 A9   18 A7   D0   02ED          MOVL    IFB$L_ARGLST(R7),IFB$L_ARGLST(R9); and arg list addr
               59   DD   02F2          PUSHL   R9                       ; save ppf ifab addr
                         02F4
                         02F4  ;
                         02F4  ;  Fill in name block, if any, with resultant string and dvi.
                         02F4  ;
                         02F4
         59    57   D0   02F4          MOVL    R7,R9                    ; restore temp ifab addr
            FD06'   30   02F7          BSBW    RM$FILLNAM               ; fill in nam blk if any
            E0 50   E9   02FA          BLBC    R0,ERRNAM                ; branch on error
      5C    0E AA   B0   02FD          MOVW    FWA$W_ESCIFI(R10),AP     ; save ifi code
            FCFC'   30   0301          BSBW    RM$DEALLOCATE_FWA        ; deallocate fwa and associated struct.
         53    59   D0   0304          MOVL    R9,R3                    ; ifab addr to r3,
         54    59   D0   0307          MOVL    R9,R4                    ;  r4 for retblk
            FCF3'   30   030A          BSBW    RM$RETBLK                ; deallocate ifab
                         030D
                         030D  ;
                         030D  ; return all pages used
                         030D  ;
                         030D
    5A    59    08  C3   030D          SUBL3   #8,R9,R10                ; get ifab list head
          54    6A  D0   0311          MOVL    (R10),R4                 ; get first hole (must be there - ifb)
                         0314
                64  DD   0314  1$:     PUSHL   (R4)                     ; save forward pointer
000001F8 8F   08 A4  D1  0316          CMPL    8(R4),#504               ; hole should be at least a page (-header)
                58  1F   031E          BLSSU   DEALLERR                 ; some one goofed
             FCDD'  30   0320          BSBW    RM$RET1PAG               ; return it
               54 8ED0   0323          POPL    R4                       ; restore pointer to next hole
          5A    54  D1   0326          CMPL    R4,R10                   ; all done?
                E9  12   0329          BNEQ    1$                       ; do next
                         032B
             FCD2'  30   032B          BSBW    RM$ZAPIFI                ; zero ifab table entry
          59    8E  D0   032E          MOVL    (SP)+,R9                 ; restore ppf ifab addr
          5A    59  D0   0331          MOVL    R9,R10                   ; copy to r10
                         0334
                         0334  ;
                         0334  ; Copy saved sp value.
```

```
                             0334  ;
                             0334
 00000014'9F    14 AB    D0  0334          MOVL    IMP$L_SAVED_SP(R11),@#PIO$GW_PIOIMPA+IMP$L_SAVED_SP
      5B   00000000'9F   3E  033C          MOVAW   @#PIO$GW_PIOIMPA,R11     ; get pio impure area addr
                             0343
                             0343  ;
                             0343  ;  Output the new ifi value and bls field to fab.
                             0343  ;
                             0343
         3C A8    0094 C9    B0  0343          MOVW    IFB$L_ASDEVBSIZ(R9),FAB$W_BLS(R8); default buffer size
                      50 8ED0  0349          POPL    R0                         ; pop return pc
                             034C
                             034C  ;
                             034C  ;  If user specified stream format but RAT=0, force RAT=CR
                             034C  ;  to get proper output.
                             034C  ;
                             034C
              04     1F A8    91  034C          CMPB    FAB$B_RFM(R8),#FAB$C_STM ; user specified stream format?
                        0B    1F  0350          BLSSU   5$                         ; branch if not
                        07    93  0352          BITB    #<FAB$M_CR!FAB$M_FTN!FAB$M_PRN>,-
                     1E A8        0354                  FAB$B_RAT(R8)              ; user specified RAT?
                        05    12  0356          BNEQ    5$                         ; branch if so
                                  0358          SSB     #FAB$V_CR,FAB$B_RAT(R8)  ; force RAT=CR
    5C     08     06     1E A8    F0  035D  5$:     INSV    FAB$B_RAT(R8),#FAB$V_PPF_RAT,#FAB$S_PPF_RAT,AP; save rat in ifi
              02 A8     5C    B0  0363          MOVW    AP,FAB$W_IFI(R8)           ; construct non-priv. ifi
                                  0367
                                  0367  ;
                                  0367  ;  Return the real rat and rfm values for the file, except
                                  0367  ;  If this is a unit record device and the user has not specified
                                  0367  ;  print file carriage control, return rfm=var and don't change the
                                  0367  ;  user's rat field.
                                  0367  ;
                                  0367
                                  0367          ASSUME  DEV$V_REC EQ 0
                                  0367
                 03 69    E8  0367          BLBS    IFB$L_PRIM_DEV(R9),20$   ; branch if unit record
                  FC93'   31  036A  10$:    BRW     RM$COPRTN1                 ; go finish up open
           F9 5C     08   E0  036D  20$:    BBS     #FAB$V_PRN+FAB$V_PPF_RAT,AP,10$; branch if user specifed
                                  0371                                             ; print file carriage ctl.
           1F A8     02   90  0371          MOVB    #FAB$C_VAR,FAB$B_RFM(R8); change rfm to var
                  FC88'   31  0375          BRW     RM$COPRTN2                 ; finish up open without
                                  0378                                             ;  altering user's rat
                                  0378
                                  0378  DEALLERR:
                                  0378          RMSTBUG FTL$_DEALLERR
                                  037F
                                  037F          .ENDC   ;STASWITCH
```

```
                                037F        .SBTTL  RM$RET_DEV_CHAR - Set device characteristics into FAB
                                037F  ;++
                                037F  ;
                                037F  ;  Set the associated device characteristics into the FAB.
                                037F  ;
                                037F  ;  The following algorithm is used to determine the characteristics:
                                037F  ;
                                037F  ;         Foreign-mounted devices:
                                037F  ;                 FAB$L_DEV = FAB$L_SDC = IFB$L_AS_DEV
                                037F  ;         Spooled devices:
                                037F  ;                 FAB$L_DEV = IFB$L_AS_DEV
                                037F  ;                 FAB$L_SDC = IFB$L_PRIM_DEV
                                037F  ;         Network devices:
                                037F  ;                 As determined by NT$REG_DEV_CHAR
                                037F  ;         Indirect PPF sequential devices:
                                037F  ;                 FAB$L_DEV = IFB$L_PRIM_DEV modified to be a record device:
                                037F  ;                         DIR, FOD, RND, SDI, SQD  cleared
                                037F  ;                         REC, CCL set
                                037F  ;                 FAB$L_SDC = IFB$L_AS_DEV
                                037F  ;         All other normal devices:
                                037F  ;                 FAB$L_DEV = IFB$L_PRIM_DEV
                                037F  ;                 FAB$L_SDC = IFB$L_AS_DEV
                                037F  ;
                                037F  ;   If device is foreign, move AS_DEV characteristics into both FAB
                                037F  ;   characteristics words, since the PRIM_DEV has been altered for
                                037F  ;   internal RMS processing requirements (e.g. DIR has been cleared, FOR
                                037F  ;   may have been set because of NFS, etc.).
                                037F  ;
                                037F  ;--
                                037F
                                037F  RM$RET_DEV_CHAR::
                                037F
      40 A8      69     D0      037F          MOVL    IFB$L_PRIM_DEV(R9),FAB$L_DEV(R8)        ; set prim dev char
   44 A8    008C C9     D0      0383          MOVL    IFB$L_AS_DEV(R9),FAB$L_SDC(R8)          ; set secondary dev char
      0A 69      18     E0      0389          BBS     #DEV$V_FOR,IFB$L_PRIM_DEV(R9),10$       ; diff if foreign
   0A 008C C9    06     E1      038D          BBC     #DEV$V_SPL,IFB$L_AS_DEV(R9),20$         ; okay if not spooled
      44 A8      69     D0      0393          MOVL    IFB$L_PRIM_DEV(R9),FAB$L_SDC(R8)        ; set alt dev. char.
   40 A8    008C C9     D0      0397  10$:    MOVL    IFB$L_AS_DEV(R9),FAB$L_DEV(R8)          ; set alt dev. char.
                                039D
                                039D          ASSUME  FAB$C_SEQ       EQ      0
                                039D
         23 A9   95             039D  20$:    TSTB    IFB$B_ORGCASE(R9)                       ; sequential file org?
            10   12             03A0          BNEQ    30$                                     ; branch if not
                                03A2
                                03A2  ;
                                03A2  ;  If this is an indirect process-permanent file, set up the device characteristics
                                03A2  ;  to make it look like a unit record device.
                                03A2  ;
                                03A2
      0C 69      22     E1      03A2          BBC     #IFB$V_PPF_IMAGE,(R9),30$               ; branch if not indirect ppf
40 A8    10004038 8F    CA      03A6          BICL2   #DEV$M_DIR!DEV$M_FOD!DEV$M_RND -        ; clear file characteristics
                                03AE                  !DEV$M_SDI!DEV$M_SQD,FAB$L_DEV(R8)
            03   C8             03AE          BISL2   #DEV$M_REC!DEV$M_CCL,-                  ; and set unit record chars.
         40 A8                  03B0                  FAB$L_DEV(R8)
                                03B2
                                03B2  ;
                                03B2  ;  Network specific code
```

```
                        03B2  1127 ;
                        03B2  1128
                        03B2  1129 30$:
                        03B2  1130          .IF     NDF,STASWITCH
                        03B2  1131
     09 69    3E    E1  03B2  1132          BBC     #IFB$V_DAP,(R9),40$              ; branch if not network oper
          23 A9     94  03B6  1133          CLRB    IFB$B_ORGCASE(R9)                ; zero orgcase to subsequent
                        03B9  1134                                                   ;  force rel and idx file op
                        03B9  1135                                                   ;  thru sequential code!!!
   00000000'EF      16  03B9  1136          JSB     NT$RET_DEV_CHAR                  ; return real device char to
                        03BF  1137
                        03BF  1138          .ENDC   ;STASWITCH
                        03BF  1139                                                   ;  if they were returned by
                    05  03BF  1140 40$:     RSB
                        03C0  1141
                        03C0  1142
                        03C0  1143          .END
```

```
$$.PSECT_EP         = 00000000          FAB$C_FIX           = 00000001
$$ARGS              = 00000004          FAB$C_SEQ           = 00000000
$$RMSTEST           = 0000001A          FAB$C_STM           = 00000004
$$RMS_PBUGCHK       = 00000010          FAB$C_UDF           = 00000000
$$RMS_TBUGCHK       = 00000008          FAB$C_VAR           = 00000002
$$RMS_UMODE         = 00000004          FAB$L_DEV           = 00000040
$$T1                = 00000000          FAB$L_FOP           = 00000004
ASSIGN                00000078 R  01    FAB$L_SDC           = 00000044
ASSIGN$_ACMODE      = 0000000C          FAB$M_CR            = 00000002
ASSIGN$_CHAN        = 00000008          FAB$M_DEL           = 00000004
ASSIGN$_DEVNAM      = 00000004          FAB$M_FTN           = 00000001
ASSIGN$_MBXNAM      = 00000010          FAB$M_GET           = 00000002
ASSIGN$_NARGS       = 00000004          FAB$M_NFS           = 00010000
CHKRND                000001D4 R  01    FAB$M_PPF_IND       = 00004000
DEALLERR              00000378 R  01    FAB$M_PRN           = 00000004
DEV$M_CCL           = 00000002          FAB$M_PUT           = 00000001
DEV$M_DIR           = 00000008          FAB$M_TRN           = 00000010
DEV$M_FOD           = 00004000          FAB$M_UFO           = 00020000
DEV$M_REC           = 00000001          FAB$M_UPD           = 00000008
DEV$M_RND           = 10000000          FAB$S_CHAN_MODE     = 00000002
DEV$M_SDI           = 00000010          FAB$S_PPF_RAT       = 00000008
DEV$M_SQD           = 00000020          FAB$V_CHAN_MODE     = 00000002
DEV$V_DIR           = 00000003          FAB$V_CR            = 00000001
DEV$V_FOR           = 00000018          FAB$V_EXE           = 00000007
DEV$V_IDV           = 0000001A          FAB$V_FTN           = 00000000
DEV$V_ODV           = 0000001B          FAB$V_GET           = 00000001
DEV$V_REC           = 00000000          FAB$V_NFS           = 00000010
DEV$V_RND           = 0000001C          FAB$V_PPF_IND       = 0000000E
DEV$V_SPL           = 00000006          FAB$V_PPF_RAT       = 00000006
DEV$V_SQD           = 00000005          FAB$V_PRN           = 00000002
DEV$V_TRM           = 00000002          FAB$V_PUT           = 00000000
DIRECTPPF             00000201 R  01    FAB$V_UFO           = 00000011
DIRECTPPF1            000000C3 R  01    FAB$W_BLS           = 0000003C
DVI$C_SECONDARY     = 00000001          FAB$W_IFI           = 00000002
DVI$_DEVBUFSIZ      = 00000008          FAB$W_MRS           = 00000036
DVI$_DEVCHAR        = 00000002          FIB$C_LENGTH        = 00000040
DVI$_DEVCLASS       = 00000004          FOP                 = 00000020
DVI$_DEVLOCKNAM     = 000000F0          FOREIGN               000001AA R  01
DVI$_FULLDEVNAM     = 000000E8          FSCB$V_ELIPS        = 00000010
DVI$_RECSIZ         = 00000018          FTL$_BADIFAB        = FFFFFFFD
DVI$_VOLNAM         = 00000022          FTL$_DEALLERR       = FFFFFFEF
ERRASGN               000000D7 R  01    FWA$B_ESCFLG        = 0000000C
ERRASGN1              000001FE R  01    FWA$B_ESCTYP        = 0000000D
ERRBG3                000002B8 R  01    FWA$C_FIBLEN        = 0000004C
ERRFOP                000002AC R  01    FWA$L_DEV_CLASS     = 0000001C
ERRIFI                000002BF R  01    FWA$L_RECSIZ        = 00000020
ERRLNE                000002B2 R  01    FWA$Q_AS_SHRFIL     = 000001A0
ERRNAM                000002DD R  01    FWA$Q_CONCEAL_DEV   = 000000E8
ERROR                 000000C2 R  01    FWA$Q_DEVICE        = 000000E0
ERRORG                000002C6 R  01    FWA$Q_DIR1          = 00000130
ERRRAT                000002D4 R  01    FWA$Q_FIB           = 00000010
ERRRFM                000002CD R  01    FWA$Q_NAME          = 00000170
EXIT                  000001F3 R  01    FWA$Q_SHRFIL        = 00000190
FAB$B_ACMODES       = 0000004A          FWA$Q_SHRFIL_LCK    = 00000198
FAB$B_FAC           = 00000016          FWA$S_AS_SHRFILBUF  = 00000010
FAB$B_RAT           = 0000001E          FWA$S_DIR_LVLS      = 00000003
FAB$B_RFM           = 0000001F          FWA$S_SHRFILBUF     = 00000010
```

```
FWA$S_SHRFIL_LCKNAM                = 00000010                      RM$PRFLNM                          00000000 RG      01
FWA$T_FIBBUF                       = 000001F4                      RM$PRFLNMALT                       0000000C RG      01
FWA$T_VOLNAM                       = 00000920                      RM$RET1PAG                         ********    X    01
FWA$V_CONCEAL_DEV                  = 00000039                      RM$RETBLK                          ********    X    01
FWA$V_DIR_LVLS                     = 0000001D                      RM$RET_DEV_CHAR                    0000037F RG      01
FWA$V_NODE                         = 00000019                      RM$SETFIB                          000000B6 RG      01
FWA$V_QUOTED                       = 0000001A                      RM$XPFN                            ********    X    01
FWA$V_WILDCARD                     = 00000018                      RM$ZAPIFI                          ********    X    01
FWA$V_WILD_DIR                     = 0000001C                      RMS$_CHN                         = 0001C0EC
FWA$W_ESCIFI                       = 0000000E                      RMS$_DIR                         = 000184CC
GETDEV_CHAR                          000000DF R       01           RMS$_FNM                         = 0001852C
IFB$B_BID                          = 00000008                      RMS$_FOP                         = 0001853C
IFB$B_FAC                          = 00000022                      RMS$_IFI                         = 00018564
IFB$B_MODE                         = 0000000A                      RMS$_LNE                         = 000185BC
IFB$B_ORGCASE                      = 00000023                      RMS$_ORG                         = 0001860C
IFB$B_RAT                          = 00000051                      RMS$_RAT                         = 0001864C
IFB$B_RFMORG                       = 00000050                      RMS$_RFM                         = 00018664
IFB$C_BID                          = 0000000B                      RMS$_WLD                         = 00018744
IFB$L_ARGLST                       = 00000018                      SETDEV                             0000022B R       01
IFB$L_ASDEVBSIZ                    = 00000094                      SETFIX                             000001E5 R       01
IFB$L_AS_DEV                       = 0000008C                      SETGET                             00000012 R       01
IFB$L_DEVBUFSIZ                    = 00000048                      SETREC                             000001F0 R       01
IFB$L_EBK                          = 00000074                      SETWRT                             00000016 R       01
IFB$L_HBK                          = 00000070                      SYS$ASSIGN                         ********   GX    01
IFB$L_PRIM_DEV                     = 00000000                      SYS$DASSGN                         ********   GX    01
IFB$V_BIO                          = 00000005                      SYS$GETDVIW                        ********   GX    01
IFB$V_BRO                          = 00000006                      TPT$L_INDPPF                       ********    X    01
IFB$V_BUSY                         = 00000020
IFB$V_CREATE                       = 00000032
IFB$V_DAP                          = 0000003E
IFB$V_PPF_IMAGE                    = 00000022
IFB$V_WRTACC                       = 00000030
IFB$W_CHNL                         = 00000020
IFB$W_LRL                          = 00000052
IFB$W_MRS                          = 00000060
IMP$C_ASYQIOEFN                    = 0000001F
IMP$L_IFABTBL                      = 00000018
IMP$L_SAVED_SP                     = 00000014
INDPPF                               0000024B R       01
IND_PPF_OK                           000002E4 R       01
NT$ASSIGN                            ********    X    01
NT$MOD_DEV_CHAR                      ********    X    01
NT$RET_DEV_CHAR                      ********    X    01
NTASGN                               000000C6 R       01
PIO$A_TRACE                          ********    X    01
PIO$GW_PIOIMPA                       ********    X    01
PSL$C_EXEC                         = 00000001
PSL$C_SUPER                        = 00000002
RESTORE_IFAB                         000002D9 R       01
RM$ASSIGN                            00000074 RG      01
RM$BUG                               ********    X    01
RM$COPRTN1                           ********    X    01
RM$COPRTN2                           ********    X    01
RM$DEALLOCATE_FWA                    ********    X    01
RM$FILLNAM                           ********    X    01
RM$GTIADR                            ********    X    01
RM$MAPERR                            ********    X    01
```

```
                                                     +----------------+
                                                     ! Psect synopsis !
                                                     +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (      0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
RMSRMS0                         000003C0  (    960.)  01 (  1.)    PIC    USR    CON    REL    GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                           00000000  (      0.)  02 (  2.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                                  +------------------------+
                                                  ! Performance indicators !
                                                  +------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                    29    00:00:00.11    00:00:00.88
Command processing               115    00:00:00.79    00:00:07.27
Pass 1                           436    00:00:17.81    00:00:50.68
Symbol table sort                  0    00:00:02.51    00:00:03.60
Pass 2                           198    00:00:04.03    00:00:11.49
Symbol table output               24    00:00:00.19    00:00:00.73
Psect synopsis output              2    00:00:00.02    00:00:00.02
Cross-reference output             0    00:00:00.00    00:00:00.00
Assembler run totals             806    00:00:25.46    00:01:14.75

The working set limit was 1800 pages.
97540 bytes (191 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1744 non-local and 37 local symbols.
1143 source lines were read in Pass 1, producing 15 object records in Pass 2.
35 pages of virtual memory were used to define 34 macros.

                                                 +--------------------------+
                                                 ! Macro library statistics !
                                                 +--------------------------+

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                     14
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                      1
_$255$DUA28:[SYSLIB]STARLET.MLB;2                  15
TOTALS (all libraries)                             30

1903 GETS were required to define 30 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RMOPRFLNM/OBJ=OBJ$:RMOPRFLNM MSRC$:RMOPRFLNM/UPDATE=(ENH$:RMOPRFLNM)+EXECML$/LIB+LIB$:RMS/LIB
```

0319 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY